JTRS Test Application (JTAP) 2.3.1

JTAP Issues

AX300249-001

REV -

16 APRIL 2004

Prepared for:

JTRS Joint Program Office (JPO)
1777 N. Kent Street, Suite 2000
Arlington, VA 22209

Prepared by:

1775 West Hibiscus Blvd, Suite 200
Melbourne, FL 32901

Document Authorization

Originator: Joe Shuhy	Date
Approved By: Ray Boggs Program Manager	Date
Approved By: Kelly Wischnefski Quality Assurance	Date
Approved By: Carrie Rather Engineering Services	Date

Document Revision History

Version/Revision	*Description of Change*	*Chg'd By*	*Chg/Rel #*	*Date*
-001/-	Initial baseline	J. Shuhy	283	16 April 2004

Table of Contents

1 SCOPE 1

1.1 *PURPOSE* 1
1.2 *INTENDED AUDIENCE* 1
1.3 *DOCUMENT OVERVIEW* 1

2 REFERENCED DOCUMENTS 2

3 JTAP ISSUES 3

3.1 *DOMAINMANAGER TESTS* 3
3.2 *DEVICEMANAGER TESTS* 4
3.3 *DEVICE TESTS* 4
3.4 *ANY QUERY TEST* 4
3.5 *COMMON TO ALL FILE TESTS* 5
3.6 *FILEMANAGER TESTS* 5
3.7 *FILESYSTEM TESTS* 5
3.8 *FILE TESTS* 6
3.9 *LOG TESTS* 6
3.10 *VARIOUS TESTS* 6
3.11 *PSEUDODEVICE CODE* 6
3.12 *HCI* 7

1 Scope

1.1 *Purpose*

The JTAP Issues document is intended to provide a summary of known issues with a specific release of JTAP. The issues may identify the specific test, or, when the problem is with a common utility, it may specify a general area where the problem may be seen.

1.2 *Intended Audience*

In order to have clear understanding of the contents of this document, the reader should be familiar and have some background of the following items:

1. Computer Software BS Degree or equivalent understanding,
2. Common Object Request Broker Architecture (CORBA),
3. Extensible Markup Language (XML),
4. C++ language,
5. The Adaptive Communication Environment (ACE) Object Request Broker (ORB) (TAO),
6. Software Communications Architecture.
7. JTAP Documentation

The reader is assumed to have a basic understanding of the architecture and functionality of the current JTRS system, and to have read and understood the Software Communications Architecture (SCA) [4][5] to include all supplements, appendices and attachments pertinent to the version of the SCA that will be tested.

1.3 *Document Overview*

This JTAP Issues document is composed of the following sections:

Section 1 - contains the purpose of this document and its intended audience.

Section 2 -contains the referenced documents.

Section 3 – contains the description of issues

2 Referenced Documents

The following specifications, standards, and handbooks are referenced within this software product specification and are applicable only to the extent specified herein.

[1]	AX300131-001	REV -	JTAP System Guide 14 August 2003.
[2]	AX300107-001	REV -	JTAP Software User Manual 14 August 2003.
[3]	AX300133-001	REV -	GUTS 2.2 Design Guide, 14 August 2003.
[4]	MSRC-5000SCA	V2.0	Software Communications Architecture Specification, 15 December 2000.
[5]	MSRC-5000SCA	V2.2	Software Communications Architecture Specification, 17 November 2001.
[6]	MSRC-5000API	V1.0	Application Program Interface Supplement to the Software Communications Architecture Specification, 15 December 2000.
[7]	MSRC-5000API	V1.1	Application Program Interface Supplement to the Software Communications Architecture Specification, 17 November 2001.
[8]	MSRC-5000SEC	V1.0	Security Supplement to the Software Communications Architecture Specification, 15 December 2000.
[9]	MSRC-5000SEC	V1.1	Security Supplement to the Software Communications Architecture Specification, 17 November 2001.

3 JTAP Issues

The issues discussed below effect JTAP 2.3.1. They are categorized by SCA component when possible. The details of JTAP test that are affected can be found in the test plans located in the GUTS 2.2 Design Guide [3]. HCI usage is described in the System Guide [1] and Software User Manual [2].

3.1 *DomainManager Tests*

1. DomainManager deviceManagers attribute passes SCA205 even when a LogService isn't found
2. DomainManager registerDeviceManager InvalidProfile might not clean up the JTAPTestDeviceManager. Also, it does not look for the FAILURE_ALARM log after it receives the exception.
3. SCA473 needs to be removed from DomainManager deviceManagers test since we cannot verify how the DeviceManager was registered.
4. There is a difference between the DomainManager Configuration Descriptor (DCD) DTD file JTAP uses for verification and Appendix D. In Appendix D the DTD does not have a "?" after the services element of the domainmanagerconfiguration element and the DTD that JTAP uses does. The "?" can be removed from after the domainmanagerconfiguration element in the domainmanagerconfiguration.2.2.dtd located under the JTAP installation directory in xml\DTD.
5. DomainManager uninstallApplication test errors may not print the correct error message.
6. DomainManager registerDeviceManager unregisterDeviceManager test installs an application that relies on the JTAP ExecutableDevice to install, but the Device is not installed until later to test a service connection requirement. This is a chicken and egg problem where both need to be installed first and the test cannot work properly.
7. Clean up at the end of a test can fail on Core Frameworks that incorrectly install the JTAP's invalid applications. This can result in an ApplicationFactory being left installed in the target system.
8. Some XML files under the JTAP installation directory in the DomainManagerXML directory are incorrect. They have a componentinstantiation id that is not a UUID. If this causes a problem for a Core Framework, then any arbitrary UUID can be inserted into those files.
9. The invalid XML used for Application installation reference an unknown fileref in the componentinstantiation element. This can be caught by a Core Framework before the intentional invalid element. The componentinstantiation fileref should refer back to the id created in the componentfile element earlier in the file.
10. The DomainManager Restore ApplicationFactory test can pass when it shouldn't. The logs may give an error message. The test plan in the GUTS 2.2 Design Guide for this test is outdated. The effect of the test is the same, the method of accomplishing it has changed.
11. Tests that verify logs can fail if the producer name is not the same as the DomainManager name registered in the Name Service. This is the logical name that the Core Framework has already assigned to the DomainManager, but it is not required to be used by logs in the SCA.

3.2 *DeviceManager Tests*

1. DeviceManager FileSystem attribute tests assumes that the object is a FileManager without checking it for NIL before using it, causing the test to fail. DeviceManager can have either a FileSystem or a FileManager.
2. In DeviceManager InvalidObject tests, an error log is missing if the exception is not thrown.
3. DeviceManager registerService unregisterService can access invalid memory if no event was received, causing the test to end before completion.
4. DeviceManager tests can report SCA474.x requirements as secondary instead of primary. They should be considered primary and treated as such.
5. The DeviceManager registeredServices attribute test fails when no services are in the DCD. The test should continue to verify that the object(s) in the sequence are not NIL and pass or fail based on the validity of those objects.
6. DeviceManager getComponentImplementationId test assumes that the DeviceManager under test installed the PseudoDevice. This won't necessarily be true.

3.3 *Device Tests*

1. Base Device and LoadableDevice tests can fail if the specified Device is not an ExecutableDevice.
2. Device tests will prompt for a DeviceManager ID. If one is supplied to the Device tests, the Device tests can fail even if the Devices under test are ExecutableDevices to work around the problem noted in issue 1 in this section.
3. If a CF service does not have an SCD file associated with it, then it is considered a device and its componentinstantiation Id is included in the devices sequence by the XML utils. This can cause DeviceManager registeredDevices attribute to fail. This is an SCA interpretation issue.
4. Device LoadFail test tries to produce an exception by using a NIL FileSystem object. A CF might throw an InvalidFileName exception since they can't find the file.
5. Device Execute exceptions that use the JTAPInvalidExecutable file will fail because of a missing file. To fix this, create a zero byte file under the JTAP installation directory in target2_2/JTAPLoadAndExceute and call it JTAPInvalidExecutable.
6. Device adminState and releaseObject tests have an issue with receiving events which will cause the tests to not receive the events and fail.
7. Device stop start test can cause an exception during clean up. The test results before the exception are valid.

3.4 *Any Query Test*

1. Query tests do not take into account struct and structseq elements as queriable properties. This is an incomplete test, and future JTAP versions will be more complete in testing.
2. Implementation specific property files are not used to extract properties. In order to use these implementation dependent propertyfiles, we have to be able to determine which implementation is currently in use.

3. Query UnknownProperty tests can fail a requirement, but still pass the test. The correct result is that if a requirement fails, the test should fail.
4. Configure InvalidConfiguration and PartialConfiguration tests can fail a requirement, but still pass the test. The correct result is that if a requirement fails, the test should fail.
5. Tests that use query properties can access invalid memory, causing the test to end before completion.
6. There are issues building PRF file paths and SCD file paths when the paths are relative to the SPD file. These tests may fail without logging an error message.

3.5 *Common to All File Tests*

1. InvalidFileName tests can print that they failed, even if they should have passed. The logs from the test can be confusing if this happens.
2. Various tests may have problems, and possibly end before completion, if the JTAP_FS_DIRECTORY environment variable is not set correctly (variable either nonexistent or pointing to an invalid directory). If JTAP is installed from one Windows account and run from another Windows account, then it is possible that the environment variable won't exist.
3. The prompt for the mountPoint name, for both the FileSystem and File tests, needs to inform the user that the mountPoint name is case-sensitive. If the test operator is unaware of the case-sensitive use, it can cause testing issues.

3.6 *FileManager Tests*

1. FileManager copy test will fail if the FileManager calls copy on the JTAP FileSystem instead of using open, create, read, and write calls. This is due to a problem in the JTAP FileSystem copy method.
2. FileManager list InvalidFileName doesn't pass the InvalidFileNames to the FileManager. Instead an empty string is passed to the FileManager.

3.7 *FileSystem Tests*

1. Minimum path length test builds a single 1024 character directory name instead of building the path by appending 40 character directory names to reach 1024 characters total.
2. Minimum file length test uses a 39 character directory name (plus a / which the author counted as the 40th character). Future versions of JTAP will have a full 40 character directory name.
3. FileSystem tests will try to run if a DeviceManager's fileSystem attribute contains a FileManager and the test operator did not specify a mountpoint parameter is empty. The tests should fail due to the missing mountpoint name and exit.
4. FileSystem list test logs a warning message if the file names returned do not contain the absolute path name in the file name. This is incorrect. A warning is to be printed if the file name is not a "simple" file name, containing no path information.
5. FileSystem query opens a file and writes to it, but doesn't close the file to ensure its contents are flushed to disk before it looks for available space to decrease. This can cause the test to fail on some operating systems that buffer data more often.

3.8 *File Tests*

1. File sizeof FileException test will fail if the errorNumber in the exception is not ENOENT. This should not be a failure, only a warning.
2. File read write close test needs to verify that when the file is opened for writing that the filePointer is positioned at the end of the file not at the beginning.
3. File sizeOf FileException will call remove on an open file. This might not work on some Core Frameworks. Even though the test method comes directly from the SCA text, it is considered an interpretation issue.

3.9 *Log Tests*

1. The Log can fail requirements and correctly report the failed requirement, but the test might not print an error statement and it may not return a failure status. These tests are considered failed. A Log test can log an error without failing the proper requirement, these are also considered failed.
2. Log getLogFullAction setLogFullAction will request logs starting with an ID that has already been overwritten. The Log Service should return an empty sequence per SCA53, but the test expects 2 valid logs and will incorrectly fail.
3. Log getAdministrativeState setAdministrativeState incorrectly uses getOperationalState instead of getAvailabilityStatus when checking if the service is off duty.
4. After writing a log, the Log tests need to delay before they check the number of logs, try to read logs, etc.

3.10 *Various Tests*

1. Various tests that look for logs, e.g. DomainManager and Application, look for specific text in the log so that it can be identified. This is difficult since there is no mandated text message in the SCA. The log can be missed if we cannot identify it as the log of interest.
2. Testing of some currently untested requirements is possible. Future versions of JTAP will have more complete test coverage.

3.11 *PseudoDevice Code*

1. The PseudoDevice code in psServer.cpp will narrow the CompositeDevice and consider it valid if an exception is not thrown. For more complete testing, it needs to make sure the narrowed object is not NIL before it considers it valid.
2. The PseudoDevice code in psServer.cpp needs the shutdown check to be changed to: *while (!PS_quit && !ps_servant->m_shutdown)* (note the && replacing ||).
3. The PseudoDevice allocateCapacity method is looking for the PD_COUNTS name instead if the UUID. To work around this, make it accept both (JTAP tests will use the name, the CF should be using the UUID).
4. The PseudoDevice IDL, and possibly others, use "Any" (uppercase A) as a CORBA data type. This has been an issue with ORBExpress users. Work around this by making it use "any" (lowercase a) instead.

3.12 *HCI*

1. Requirements Tree sorting preference isn't getting initialized when JTAP starts. This leaves it vulnerable to starting in the very slow "sort by status" mode when the operator actually wants the faster "sort by name".
2. Due to SCA wording, some CF developers put a “/” into the name service instead if using it for context parsing. JTAP was modified to accommodate this, which means that JTAP currently does not parse a context. Instead it uses the context name as a literal name in the name service. If your context is not using a simple DomainName context with no extension and a DomainManager context with no extension, then you will have issues running the current version of JTAP.
3. Bug reporting from JTAP points to an Aeronix website. The new location for reporting issues is: https://jtel.spawar.navy.mil/pcr.asp